Introduction

There is no point in reinventing the wheel. So instead of explaining all the possible vectors here it might be easier to use the OWASP Mobile Testing framework.

OWASP has provided two resources that are very useful.

1. OWASP Mobile Security Testing Guide
2. Mobile App Security Checklist

   The guide and the checklist goes hand in hand. And the checklist refers to the guide quite often. The guide and the checklist concerns both Android and iPhone.

Setup

Android

We wan to do three different things to test the application: static analysis, dynamic analysis and api-analysis.

Static analysis simply refers to inspecting the contents of the application package. In android this is an apk-file. The static analysis can be done using MobSF in linux. A dynamic analysis refers to analysis of the package when it is running. To analyse that we will use Drozer.

API-test - Configuration

What we want to do is to be able to send the mobile applications traffic to burp to inspect the http traffic. This can be done in two ways. By proxying the phones traffic to your burp instance, or set up an access point and MITM the phone that way.

Invisible intercept

So this technique works by creating a access-point that your mobile phone connects to, and then man-in-the-middle all the traffic through iptables.

- Redirect the network-card to your VM

  The first problem I ran into was that the USB-adapter wouldn't be redirected into the virtual machine. I looked in syslog when I tried to redirect it, and I noticed some error messages from apparmor. It was something like this:

```
apparmor="DENIED" operation="open" profile="libvirt-..." name="/run/udev/data/+usb" comm="qemu-system-x86"
```

The command was `qemu-system-x86`, and the operation was `open`, but apparmor denied the action. After some googeling I figured out that I need to edit the apparmor profile for libvirt-qemu. So I edited this file like this:

```
sudo vim /etc/apparmor.d/abstractions/libvirt-qemu
```

```
# For hostdev access. The actual devices will be added dynamically
/sys/bus/usb/devices/ r,
/sys/devices/**/usb[0-9]*/** r,

# THIS LINE NEEDS TO BE ADDED
/run/udev/data/* rw,
```

- Add USB to Virtual Machine

If the USB adapter is using USB 3 you might need to change the USB Controller in virt-manager to USB 3. Just change it in Controller USB from 2 to 3. Next you go to `Add Hardware`, `USB Host Device`. Another way is to open the VM in spicy/virt-manager and then go to Input/Select USB Devices for Redirection. That usually doesn't work for me though.

Verify that the USB-adapter is connected by either checking `ifconfig` or `lsusb`. Hopefully you will see the wlan0 network interface.

- Create a hotspot

If you have network-manager it is really simple to set up an access-point. Open up network-manager:

```
nm-connection-editor
```

Then you click `Add` select `Wi-fi`. You name the hotspot and change `Mode` from `Client` to `Hotspot`. Then you edit wifi-security, and click `Save`.

Now you should have a working hotspot. You can verify it by looking for it from your host or smartphone. Now just connect to it from the smartphone you want to MITM.

You should now be able to see the phones traffic passing by. You can use tcpdump to see that it works.

If your device connects but does not get an IP it is probably because you are missing a dhcp-server. You just need to install dnsmasq, and network manager will figure out the rest.

```
tcpdump -i wlan0
```

- Redirect traffic to Burp through iptables

```
apt-get install iptables-persistent

iptables-save > rules

# Then you edit the rules and add them by:

iptables-restore < rules
```

So these are the rules to be added. The ip-address is the address for the smartphone that we want to intercept.

```
:PREROUTING ACCEPT [414:27612]
-A PREROUTING -i wlan0 -s 10.41.1.121 -p tcp -m tcp --dport 443 -j REDIRECT --to-ports 8080
-A PREROUTING -i wlan0 -s 10.42.0.121 -p tcp -m tcp --dport 80 -j REDIRECT --to-ports 8080
-A PREROUTING -i wlan0 -s 10.42.0.121 -p tcp -m tcp --dport 53 -j REDIRECT --to-ports 53


:POSTROUTING ACCEPT [289:20303]
-A  POSTROUTING -j MASQUERADE
```

- Configure BURP

Now you just need to configure burp.

go to `Proxy/options/Proxy Listener/Add/Choose your interface and port`. Then Click on `Request Handeling` and then `Support invisible proxying`.

- Import Burp certificate

Now you need to import you burp-certificate to you phone, in order to handle HTTPS traffic.

Remember that this will not work if the application you are assessing is using certificate pinning.

Open the web browser in your phone and go to your machines ip-address on port 8080 (or whatever port burp is listening on).

Download the burp-certificate. Rename it to `cacert.cer`

Go to `settings`, `security`, `install from storage`. And then select your cerrificate.

- Get the application on to the phone.

This is usually a really simple process. Just store the apk on your attacking-machine and have the phone download it through a webserver

```
python -m SimpleHTTPServer 80
```

Use proxy-settings on your phone

The other way of doing it is simply to set up your hotspot, and then connect the phone onto the hotspot and then configure the phone to go tospecific proxy settings.

So you go to Settings/wifi/(hold on the specified network/modify network/proxy/manual. Scroll down and then input the ip address of your hotspot, for example 10.42.0.1 and the proxy port: 8080.

Now you need to configure burp to listen to your wifi interface. Proxy/Options/Proxy listener

Installing drozer

Download drozer here: https://github.com/mwrlabs/drozer/releases

```
sudo dpkg -i drozer.deb
sudo apt install -f
```

Install drozer on phone. Start it. Check the settings that a password is set.